24-678: Computer Vision for Engineers

Carnegie Mellon University

PS1

Due:	**9/17/2021 (Fri) 5:00PM @ BOX**
Issued:	**9/8/2021 (Wed)**
Weight:	**5% of total grade**
Note:	**Welcome to 24-678: Computer Vision for Engineers!**

PS1-1 Let's get started! (Note: No need to hand in anything for this problem.)

(1) Class webpage

Carefully read the class webpage, http://www.andrew.cmu.edu/course/24-678/, and let the instructor know if you have questions about the course content, grading criteria, or schedule.

(2) Gradescope

You will be submitting your work on Gradescope. Your account will be created by 9/13 (Mon).

(3) Programming in Python

If you have taken a course on programming in Python, review your old textbook and refresh your memory on how to write Python code.

(4) Anaconda and OpenCV

Install Anaconda on your computer, and use Conda to create a virtual environment for your OpenCV problem sets. Run the two pieces of test code introduced in class on 9/1 (Wed). The sample code can be found in a handout BOX folder: https://cmu.box.com/s/tika1o2f2xb18h1vjz2fxkbo0m758v2o.

PS1-2 Read color images, apply thresholding, and change colors

Using OpenCV, write a Python program that takes as input a color image and paints bright/dark regions with red to emphasize them. For example, the program reads an image of a circuit board, "circuit_board.png," shown in Figure 1(a), and generates another color image shown in Figure 1(d) by painting the wiring with red and keeping the background unchanged.

Use this five-step method to achieve this color conversion:

(1) Ask the user for an input color image and whether the program should emphasize brighter regions or darker regions. Display the input image in the first window (see Figure 1(a)).

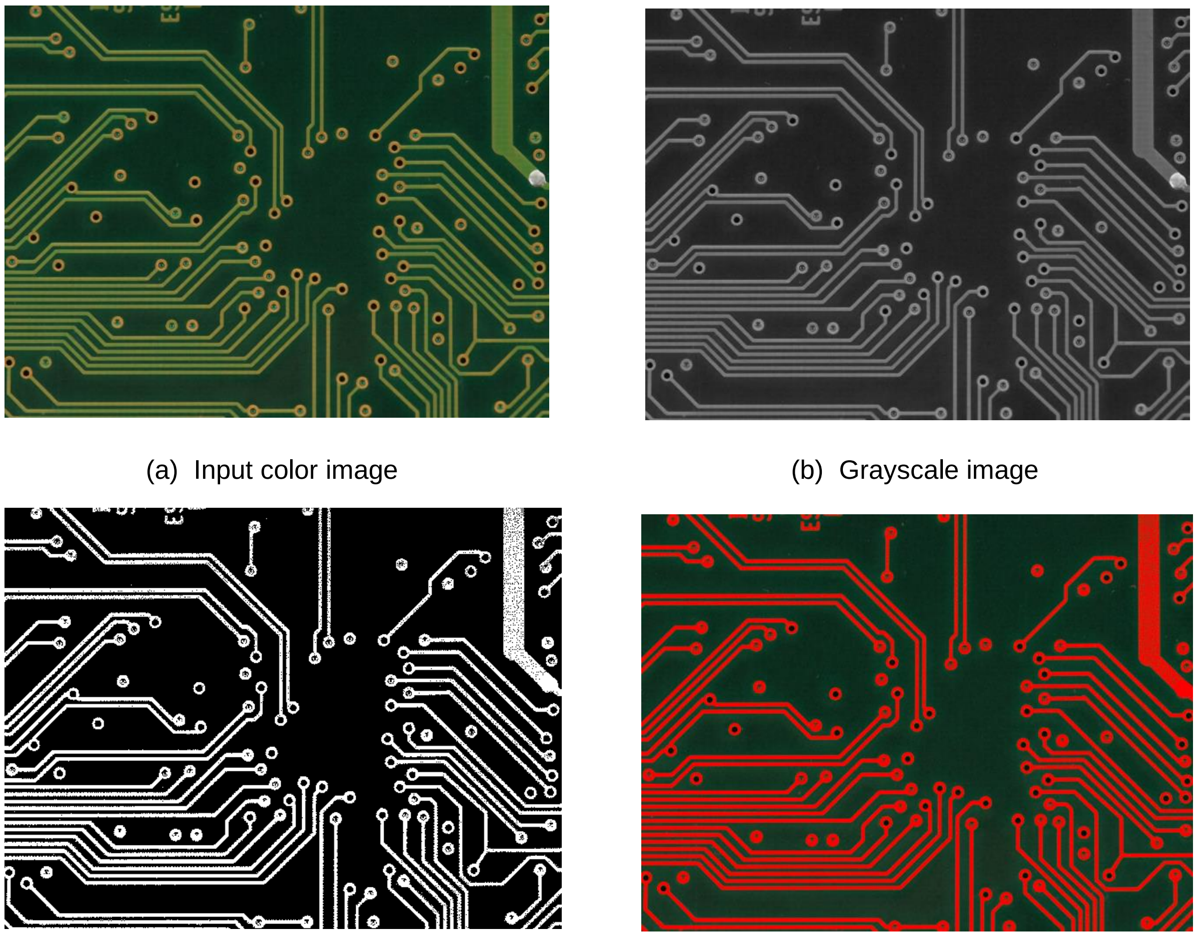
(2) Convert the color image to a grayscale image and display the image in the second window (see Figure 1(b)); save and name and the grayscale image file by adding "_grayscale" to the input filename.

(3) Convert the grayscale image to a binary image by using a threshold value that can differentiate the brighter regions and darker regions. Display the image in the third window (see Figure 1(c)); save and name the binary image file by adding "_binary" to the input filename.

(4) Create the output color image by painting each pixel of the wiring with red. Display the image in the fourth window (see Figure 1(d)); save and name the output image file by adding "_output" to the input filename.

For "circuit.png," shown in Figure 1, we want to emphasize wiring regions, or brighter regions. On the other hand, for "carnival.png," shown in Figure 2, we want to emphasize crack regions, or darker regions.

Your program should prompt the user to specify the filename of an input image, whether the program should emphasize brighter regions or darker regions, display four images (input, grayscale, binary, and output) on the screen, and save the grayscale, binary, and output image files in the same directory. Make sure to include plenty of comment lines in your Python code.



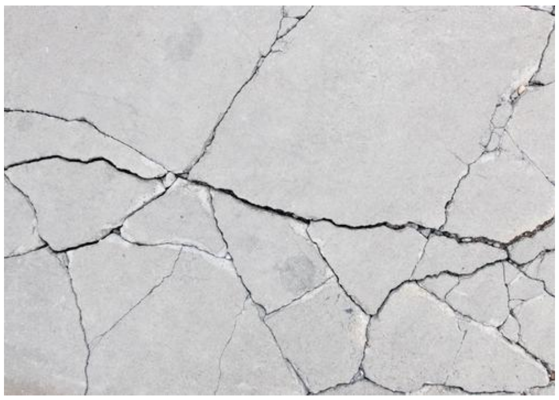
(a) Input color image

(b) Grayscale image

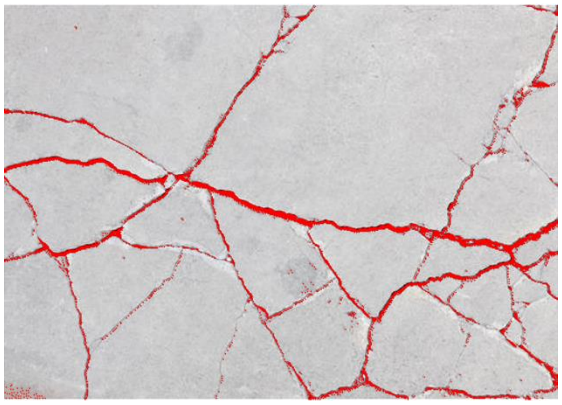
(c) Black-and-white image

(d) Output color image

Figure 1. Image color conversion – circuit.png



(a) Input color image



(b) Output color image

Figure 2. Image color conversion – crack.png

Submission

To prepare for the submission of your work on Gradescope, create:

(1) a folder called "ps1-2," that contains the following files:

- source code file(s)
- grayscale image files: "circuit_grayscale.png" and "crack_grayscale.png"
- binary image files: "circuit_binary.png" and "crack_binary.png"
- output image files: "circuit_output.png" and "crack_output.png"
- "readme.txt" file that includes:
    - Operating system
    - IDE you used to write and run your code
    - The number of hours you spent to finish this problem

(2) a PDF file that contains the printouts and screenshots of all the files in the ps1-2 folder. (Include, if any, the mathematical derivation and/or description of your method in the PDF file. Handwritten notes should be scanned and included in the PDF file.)

PS1-3 Gamma correction

Using OpenCV, write a Python program that takes as input a color image and applies gamma correction to make the image more natural-looking and/or appealing. Use your program to find the gamma value that makes an image as natural-looking and/or appealing as possible.

Your program should:

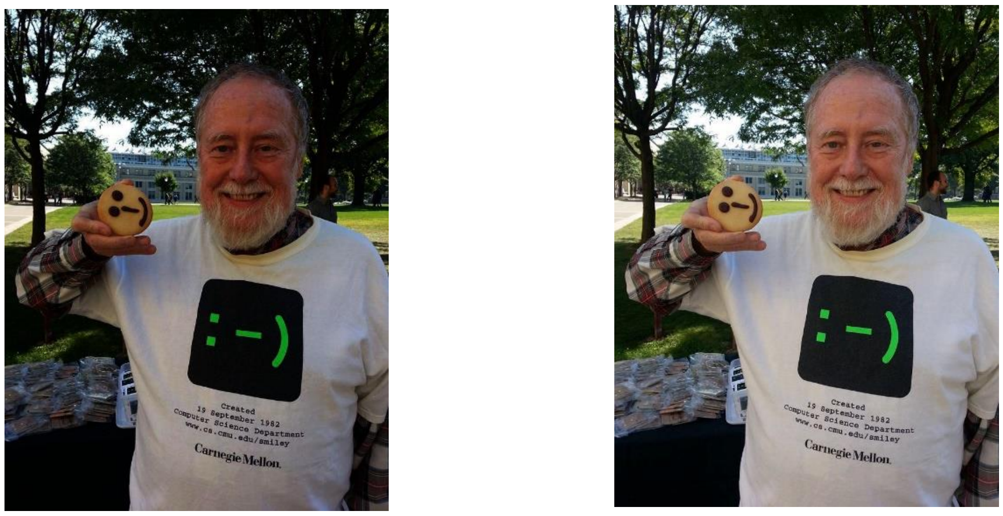
(1) Ask the user for an input color image and open two image display windows, one for the original image (Window 1) and the other for a gamma-corrected image (Window 2),

(2) Ask the user to specify a gamma value and show the corrected image in Window 2,

(3) Repeat Step 2 iteratively until you find the best gamma value, and

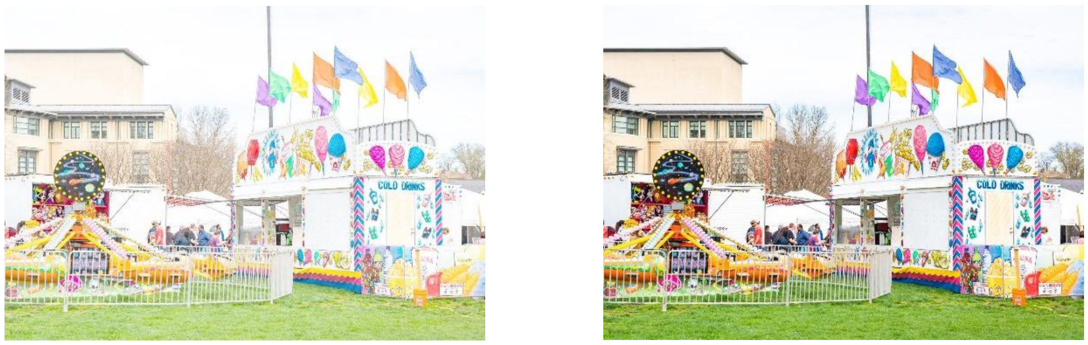
(4) Save the final corrected image file; name the file by adding "_gcorrected" to the input file name.

Make sure to include plenty of comment lines in your Python code. Apply your program to two files, smiley.jpg and carnival.jpg, shown in Figures 3 and 4.



(a) Input image (b) Output image after gamma correction

Figure 3. Gamma correction – smiley.jpg



(a) Input image (b) Output image after gamma correction

Figure 4. Gamma correction – carnival.jpg

<u>Submission</u>

To prepare for the submission of your work on Gradescope, create:

(1) a folder called “ps1-3,” that contains the following files:

- source code file(s)
- gamma-corrected images: “smiley_gcorrected.jpg” and “carnival_gcorrected.jpg”
- "readme.txt" file that includes:
    - Operating system
    - IDE you used to write and run your code
    - The number of hours you spent to finish this problem

(2) a PDF file that contains the printouts and screenshots of all the files in the ps1-3 folder. (Include, if any, the mathematical derivation and/or description of your method in the PDF file. Handwritten notes should be scanned and included in the PDF file.)

Submit your work on Gradescope

Submit two files on Gradescope – replace “andrewid” with your own Andrew ID:

(1) **andrewid-ps1-files.zip** – this ZIP file should contain two folders, “ps1-2” and “ps1-3,” and all the files requested in PS1-2 and PS1-3.

Please make sure that files are organized in two separate folders in the ZIP file: “ps1-2” and “ps1-3.”

(2) **andrewid-ps1-report.pdf** – this PDF file serves as the report of your work, and it should contain the printouts and screenshots of all the files in the “ps1-2” folder and “ps1-3” folder. (Include, if any, the mathematical derivation and/or description of your method in the PDF file. Handwritten notes should be scanned and included in the PDF file.)

Please organize pages with section titles and captions to make the report easy to read.